SAVI
Internet-Draft
Intended status: Informational
Expires: December 22, 2011

C. An
J. Yang
J. Wu
J. Bi
CERNET

June 20, 2011

Definition of Managed Objects for SAVI Protocol
draft-an-savi-mib-01

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it defines objects for managing SAVI (Source Address Validation Improvements) protocol instance.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 22, 2011.

Copyright Notice

Table of Contents

1. Introduction

The Source Address Validation Improvement protocol was developed to complement ingress filtering with finer-grained, standardized IP source address validation(refer to [I-D.ietf-savi-framework]).A SAVI protocol instance is located on the path of hosts' packets, enforcing the hosts' use of legitimate IP source addresses.

SAVI protocol determines whether the IP address obtaining process is legitimate according to IP address assignment method. For links with Stateless Address Auto Configuration (SLAAC), Dynamic Host Configuration Protocol (DHCP), and Secure Neighbor Discovery (SEND), the process is defined in separate documents of SAVI Working Group (refer to [I-D.ietf-savi-dhcp], [I-D.ietf-savi-fcfs], [I-D.ietf-savi-send].)

This document defines a MIB module that can be used to manage the SAVI protocol instance. It covers both configuration and status monitoring aspects of SAVI implementations.

This document uses terminology from the SAVI Protocol specification.

2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

3. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

4. Overview

The SAVI Protocol MIB module (SAVI-MIB) is conformant to SAVI protocol, and is designed to:

- Support centralized management and monitoring of SAVI protocol instance by standard SNMP protocol.
- Support configuration and querying of SAVI protocol parameters.
- Support configuration and querying of binding entries. Operators may insert and delete static binding entries.
- Support querying of filtering entries.

Based on SAVI protocol, attributes and objects of a SAVI protocol instance can be classified into four categories:

- System attributes. These attributes are corresponding to a SAVI protocol instance, such as IP Address Assignment Methods and some constants.
- Anchor attributes. These attributes are corresponding to a SAVI anchor. Anchor is defined in [I-D.ietf-savi-framework].
- Binding Status Table. This table contains the state of binding between source address and binding anchor (refer to [I-D.ietf-savi-dhcp]).
- Filtering Table. This table contains the bindings between binding anchor and address, which is used to filter packets (refer to [I-D.ietf-savi-dhcp]).

A table is designed for each category of objects.

5. Structure of the MIB Module

This section presents the structure of the SAVI-MIB module. The MIB objects are derived from the SAVI protocol specification.

This MIB is composed of a series of tables meant to form the base for managing SAVI entities. The following subsections describe all tables in the SAVI MIB module.

5.1. The SAVI System Table

The SAVI System Table (saviObjectsSystemTable) contains the objects which are corresponding to SAVI system-wide parameters. It supports the configuration and collection of SAVI system-wide parameters.

There is an entry for each IP stack, IPv4 and IPv6. The table is indexed by:

- saviObjectsSystemIPVersion - The IP Version. A textual convention InetVersion defined in RFC4001 is used to represent the different version of IP protocol.

It contains the following objects:

- saviObjectsSystemMode - Which IP address assignment method the link is running in (refer to [I-D.ietf-savi-framework]).

- saviObjectsSystemMaxDadDelay - A constant defined in SAVI protocol (refer to [I-D.ietf-savi-dhcp]).

- saviObjectsSystemMaxDadPrepareDelay - A constant defined in SAVI protocol (refer to [I-D.ietf-savi-dhcp]).

The MAX-ACCESS of thses objects is READ-WRITE. Network Operators may do configuration by setting these objects.

5.2. The SAVI Interface Table

The SAVI Interface Table (saviObjectsIfTable) contains the objects which are corresponding to SAVI running parameters of each anchor. It supports the configuration and collection of SAVI parameters of each anchor.

There is an entry for each IP stack, IPv4 and IPv6. The table is indexed by:

- saviObjectsIfIPVersion - The IP Version.

- saviObjectsIfIfIndex - The index value that uniquely identifies the interface to which this entry is applicable.

It contains the following objects:

- saviObjectsIfValidationStatus - The validation status of the interface. enable(1): check source address. disable(2): don’t check source address.

- saviObjectsIfTrustStatus - The trust status of the interface.

- saviObjectsIfFilteringNum - The max filtering number of the Interface.

The MAX-ACCESS of thses objects is READ-WRITE. Network Operators may configure by setting these objects.

5.3. The SAVI Binding Table

The SAVI Binding Table (saviObjectsBindingTable) contains the objects which are corresponding to Binding State Table (BST) defined in SAVI protocol. It contains the binding parameters and state of each binding entry. It supports the collection of binding entries. And an entry can be inserted or deleted if it is a static binding entry.

The table is indexed by:

- saviObjectsBindingIpAddressType - IP address type. A textual convention InetAddressType defined in RFC4001 is used to represent the different kind of IP address.
- saviObjectsBindingType - which IP address assignment method is used to create the binding entry - static(1), slaac(2), dhcp(3).
- saviObjectsBindingIfIndex - The index value that uniquely identifies the interface to which this entry is applicable.
- saviObjectsBindingIpAddress - The binding source IP address. A textual convention InetAddress defined in RFC4001 is used to define this object.

The SAVI Binding Table contains the following objects:

- saviObjectsBindingMacAddr - The binding source mac address.
- saviObjectsBindingState - The state of the binding entry.
- saviObjectsBindingLifetime - The remaining lifetime of the entry.
- saviObjectsBindingRowStatus - The status of this row, by which new entries may be created, or old entries be deleted from this table. As defined in RFC2579, the RowStatus textual convention is used to manage the creation and deletion of conceptual rows. For SAVI Binding Table, an entry can be created or deleted only when saviObjectsBindingType=static.

The MAX-ACCESS of thses objects is READ-CREATE. Network Operators may create or delete an entry by setting these objects.

5.4. The SAVI Filtering Table

The SAVI Filtering Table (saviObjectsFilteringTable) contains the objects which are corresponding to Filtering Table (FT) defined in SAVI protocol. It supports the collection of filtering entries.

The table is indexed by:

- saviObjectsFilteringIpAddressType - IP address type.
- saviObjectsFilteringIfIndex - The index value that uniquely identifies the interface to which this entry is applicable.
- saviObjectsFilteringIpAddress - The source IP address.

It contains the following objects:

- saviObjectsFilteringMacAddr - The source mac address.

The MAX-ACCESS of the object is READ-ONLY.

6. Textual Conventions

The textual conventions used in the SAVI-MIB are as follows.

The MODULE-COMPLIANCE,OBJECT-GROUP textual convention is imported from SNMPv2-CONF [RFC2580]. The MODULE-IDENTITY, OBJECT-IDENTITY, OBJECT-TYPE, Unsigned32 textual convention is imported from SNMPv2-SMI [RFC2578].

The MacAddress,TimeInterval,RowStatus textual convention is imported from SNMPv2-TC [RFC2579].

The InetVersion,InetAddressType,InetAddress textual convention is imported from INET-ADDRESS-MIB [RFC4001].

The InterfaceIndex textual convention is imported from IF-MIB [RFC2863].

The ip textual convention is imported from IP-MIB [RFC4293].

7. Relationship to Other MIB Modules

7.1. Relationship to the INET-ADDRESS-MIB

To support extensibility, IETF defined new textual conventions to represent different IP protocol and different IP address in a unified formation in RFC4001. To support different IP version, a textual convention InetVersion is defined to represent the different version of IP protocol. To support different IP address, a generic Internet address is defined. It consists of two objects: The first one has the syntax InetAddressType, and the second object have the syntax InetAddress. The value of the first object determines how the value of the second is encoded.

Since SAVI running mode and parameter is independent of IPv4 and IPv6, so different OID instances should be defined for each protocol. In SAVI-MIB definition, when IP address is used as a part of binding table, it is defined using textual conventions described in INET-ADDRESS-MIB.

7.2. Relationship to the IF-MIB

The Interfaces MIB [RFC2863] defines generic managed objects for managing interfaces. This document contains the interface-specific extensions for managing SAVI anchors that are modeled as interfaces.

The IF-MIB module is required to be supported on the SAVI device. The interface MUST be modeled as an ifEntry, and ifEntry objects such as ifIndex are to be used as per [RFC2863].

An ifIndex [RFC2863] is used as a common index for interfaces in the SAVI-MIB modules.

7.3. MIB modules required for IMPORTS

The SAVI MIB module IMPORTS objects from SNMPv2-SMI [RFC2578], SNMPv2-TC [RFC2579],SNMPv2-CONF [RFC2580], IF-MIB [RFC2863] and INET-ADDRESS-MIB [RFC4001] .

8. Definitions

```
SAVI-MIB DEFINITIONS ::=BEGIN

IMPORTS
    MODULE-COMPLIANCE,OBJECT-GROUP
        FROM SNMPv2-CONF                    --RFC2580
    MODULE-IDENTITY, OBJECT-IDENTITY, OBJECT-TYPE, Unsigned32
        FROM SNMPv2-SMI                     --RFC2578
    TEXTUAL-CONVENTION,MacAddress,TimeInterval,RowStatus
        FROM SNMPv2-TC                      --RFC2579
    InterfaceIndex
        FROM IF-MIB                         --RFC2863
    InetVersion,InetAddressType,InetAddress
        FROM INET-ADDRESS-MIB               --RFC4001
    ip
        FROM IP-MIB                         --RFC4293
    ;

saviMIB MODULE-IDENTITY
    LAST-UPDATED "201106200037Z" --June 20,2011
    ORGANIZATION
        "IETF SAVI Working Group"
```

```
    CONTACT-INFO
        "WG charter:
           http://datatracker.ietf.org/wg/savi/charter/

         Editor:
           Changqing An
           CERNET
           Postal:  Network Research Center, Tsinghua University
           Beijing 100084
           China
           Email: acq@cernet.edu.cn

           Jiahai Yang
           CERNET
           Postal:  Network Research Center, Tsinghua University
           Beijing 100084
           China
           Email: yang@cernet.edu.cn
        "

    DESCRIPTION
        "This MIB Module is designed to support configuration
        and monitoring of SAVI  protocol.
        "
    REVISION    "201003080037Z"
    DESCRIPTION
        "Initial version"
    ::= {ip xxx}

saviObjects OBJECT IDENTIFIER ::= { saviMIB 1 }

-- System parameters for SAVI protocol

saviObjectsSystemTable OBJECT-TYPE
    SYNTAX     SEQUENCE OF SaviObjectsSystemEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "The table containing savi system-wide parameters."
    ::= { saviObjects 1 }

saviObjectsSystemEntry OBJECT-TYPE
    SYNTAX     SaviObjectsSystemEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "An entry  containing savi system-wide parameters for a
            particular IP version.
```

```
            "
    INDEX { saviObjectsSystemIPVersion  }
    ::= { saviObjectsSystemTable 1 }

SaviObjectsSystemEntry ::=
    SEQUENCE {
        saviObjectsSystemIPVersion             InetVersion,
        saviObjectsSystemMode                  INTEGER,
        saviObjectsSystemMaxDadDelay           TimeInterval,
        saviObjectsSystemMaxDadPrepareDelay    TimeInterval
    }

saviObjectsSystemIPVersion OBJECT-TYPE
    SYNTAX     InetVersion
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "The IP version "
    ::= { saviObjectsSystemEntry 1 }

saviObjectsSystemMode OBJECT-TYPE
    SYNTAX     INTEGER {
                 savi-disable(1),
                 savi-default(2),
                 savi-dhcp-only(3),
                 savi-slaac-only(4),
                 savi-dhcp-slaac-mix(5),
                 savi-send(6)
               }
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
           "IP Address Assignment Methods. "
    ::= { saviObjectsSystemEntry 2 }

saviObjectsSystemMaxDadDelay OBJECT-TYPE
    SYNTAX     TimeInterval
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
           "A constant. When A gratuitous ARP Request or Duplicate
            Address Detection Neighbor Solicitation is received
            from anchor, the lifetime of the BST(Binding State Table)
            entry MUST be set to be MAX_ARP_DELAY or MAX_DAD_DELAY
            respectively.
            TimeInterval is defined in RFC 2579, it's a period of time,
            measured in units of 0.01 seconds,
            and the value is (0..2147483647).
```

```
            "
    ::= { saviObjectsSystemEntry 3 }

saviObjectsSystemMaxDadPrepareDelay OBJECT-TYPE
    SYNTAX     TimeInterval
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
           "A constant. When a DHCPv4 DHCPACK or DHCPv6 REPLY message
            is received, the lifetime of the BST(Binding State Table)
            entry MUST be set to be MAX_ARP_PREPARE_DELAY or
            MAX_DAD_PREPARE_DELAY respectively.
            TimeInterval is defined in RFC 2579, it's a period of time,
            measured in units of 0.01 seconds,
            and the value is (0..2147483647).
           "
    ::= { saviObjectsSystemEntry 4 }


-- Interface parameters for SAVI protocol

saviObjectsIfTable OBJECT-TYPE
    SYNTAX     SEQUENCE OF SaviObjectsIfEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "The table containing SAVI parameters of each anchor."
    ::= { saviObjects 2 }

saviObjectsIfEntry OBJECT-TYPE
    SYNTAX     SaviObjectsIfEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "An entry containing SAVI running parameters of an anchor."
    INDEX {
            saviObjectsIfIPVersion,
            saviObjectsIfIfIndex
          }
    ::= { saviObjectsIfTable 1 }

SaviObjectsIfEntry ::=
    SEQUENCE {
        saviObjectsIfIPVersion          InetVersion,
        saviObjectsIfIfIndex            InterfaceIndex,
        saviObjectsIfValidationStatus   INTEGER,
        saviObjectsIfTrustStatus        INTEGER,
        saviObjectsIfFilteringNum       Unsigned32
```

```
    }

saviObjectsIfIPVersion      OBJECT-TYPE
    SYNTAX     InetVersion
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "The IP version "
    ::= { saviObjectsIfEntry 1 }

saviObjectsIfIfIndex  OBJECT-TYPE
    SYNTAX     InterfaceIndex
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "The index value that uniquely identifies the interface to
            which this entry is applicable.  The interface identified by
            a particular value of this index is the same interface as
            identified by the same value of the IF-MIB's ifIndex.
           "
    ::= { saviObjectsIfEntry 2 }

saviObjectsIfValidationStatus OBJECT-TYPE
    SYNTAX     INTEGER {
                 enable(1),
                 disable(2)
               }
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
           "The validation status of the interface.
            enable(1), check source address.
            disable(2), don't check source address.
           "
    ::= { saviObjectsIfEntry 3 }

saviObjectsIfTrustStatus OBJECT-TYPE
    SYNTAX     INTEGER  {
                 no-trust(1),
                 dhcp-trust(2),
                 ra-trust(3),
                 savi-savi(4)
               }
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
          "The trust status of the interface.
           no-trust(1), discard dhcp adv/reply and ra packet.
```

```
            dhcp-trust(2), permit DHCP adv/relay packet.
            ra-trust(3), permit ra packet.
            savi-savi(4), permit any packet.
           "
    ::= { saviObjectsIfEntry 4 }

saviObjectsIfFilteringNum OBJECT-TYPE
    SYNTAX     Unsigned32
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
           "The max filtering number of the Interface."
    ::= { saviObjectsIfEntry 5 }


-- Binding Status Table for SAVI protocol

saviObjectsBindingTable OBJECT-TYPE
    SYNTAX     SEQUENCE OF SaviObjectsBindingEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "The table containing the state of binding
            between source address and anchor.
           "
    ::= { saviObjects 3 }

saviObjectsBindingEntry OBJECT-TYPE
    SYNTAX     SaviObjectsBindingEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "An entry containing the state of binding between source
            address and anchor.
            Entries are keyed on the source IP address type,
            binding type, anchor, and source IP address.
           "
    INDEX {
            saviObjectsBindingIpAddressType,
            saviObjectsBindingType,
            saviObjectsBindingIfIndex,
            saviObjectsBindingIpAddress
          }
    ::= { saviObjectsBindingTable 1 }

SaviObjectsBindingEntry ::=
    SEQUENCE {
        saviObjectsBindingIpAddressType  InetAddressType,
```

```
        saviObjectsBindingType          INTEGER,
        saviObjectsBindingIfIndex       InterfaceIndex,
        saviObjectsBindingIpAddress     InetAddress,
        saviObjectsBindingMacAddr       MacAddress,
        saviObjectsBindingState         INTEGER,
        saviObjectsBindingLifetime      TimeInterval,
        saviObjectsBindingRowStatus     RowStatus
     }

saviObjectsBindingIpAddressType OBJECT-TYPE
    SYNTAX     InetAddressType
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "IP address type of the  binding source IP."
    ::= { saviObjectsBindingEntry 1 }

saviObjectsBindingType OBJECT-TYPE
    SYNTAX     INTEGER {
                 static(1),
                 slaac(2),
                 dhcp(3)
               }
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "IP address assignment methods."
    ::= { saviObjectsBindingEntry 2 }

saviObjectsBindingIfIndex OBJECT-TYPE
    SYNTAX     InterfaceIndex
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "The index value that uniquely identifies the interface to
            which this entry is applicable.  The interface identified by
            a particular value of this index is the same interface as
            identified by the same value of the IF-MIB's ifIndex.
           "
    ::= { saviObjectsBindingEntry 3 }

saviObjectsBindingIpAddress OBJECT-TYPE
    SYNTAX     InetAddress
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "The binding source IP address"
    ::= { saviObjectsBindingEntry 4 }
```

```
saviObjectsBindingMacAddr OBJECT-TYPE
    SYNTAX     MacAddress
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
           "The binding source mac address."
    ::= { saviObjectsBindingEntry 5 }

saviObjectsBindingState OBJECT-TYPE
    SYNTAX     INTEGER  {
                   start(1),
                   live(2),
                   detection(3),
                   query(4),
                   bound(5)
               }
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
           "The state of the binding entry. "
    ::= { saviObjectsBindingEntry 6 }

saviObjectsBindingLifetime OBJECT-TYPE
    SYNTAX     TimeInterval
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
           "The remaining lifetime of the entry.
            TimeInterval is defined in RFC 2579, it's a period of time,
            measured in units of 0.01 seconds,
            and the value is (0..2147483647).
            If saviObjectsBindingType=static, a value of 2147483647
            represents infinity.
           "
    ::= { saviObjectsBindingEntry 7 }

saviObjectsBindingRowStatus OBJECT-TYPE
       SYNTAX     RowStatus
       MAX-ACCESS read-create
       STATUS     current
       DESCRIPTION
             "The status of this row, by which new entries may be
              created, or old entries deleted from this table.
              An Entry can be created or deleted only when
              saviObjectsBindingType=static.
             "
    ::= { saviObjectsBindingEntry 8 }
```

```
-- Filtering Table for SAVI protocol

saviObjectsFilteringTable OBJECT-TYPE
    SYNTAX     SEQUENCE OF SaviObjectsFilteringEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "The table containing the filtering entries."
    ::= { saviObjects 4 }

saviObjectsFilteringEntry OBJECT-TYPE
    SYNTAX     SaviObjectsFilteringEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "An entry containing the filtering parameters.
            Entries are keyed on the source IP address type,
            anchor, and source IP address.
           "
    INDEX { saviObjectsFilteringIpAddressType,
            saviObjectsFilteringIfIndex,
            saviObjectsFilteringIpAddress
          }
    ::= { saviObjectsFilteringTable 1 }

SaviObjectsFilteringEntry ::=
    SEQUENCE {
        saviObjectsFilteringIpAddressType  InetAddressType,
        saviObjectsFilteringIfIndex        InterfaceIndex,
        saviObjectsFilteringIpAddress      InetAddress,
        saviObjectsFilteringMacAddr        MacAddress
    }

saviObjectsFilteringIpAddressType OBJECT-TYPE
    SYNTAX     InetAddressType
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "IP address type of the  filtering source IP"
    ::= { saviObjectsFilteringEntry 1 }

saviObjectsFilteringIfIndex OBJECT-TYPE
    SYNTAX     InterfaceIndex
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "The index value that uniquely identifies the interface to
            which this entry is applicable.  The interface identified by
```

```
            a particular value of this index is the same interface as
            identified by the same value of the IF-MIB's ifIndex.
           "
    ::= { saviObjectsFilteringEntry 2 }

saviObjectsFilteringIpAddress OBJECT-TYPE
    SYNTAX     InetAddress
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
           "The filtering source IP address."
    ::= { saviObjectsFilteringEntry 3 }

saviObjectsFilteringMacAddr OBJECT-TYPE
    SYNTAX     MacAddress
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
           "The filtering source mac address."
    ::= { saviObjectsFilteringEntry 4 }

-- Conformance information
saviConformance OBJECT IDENTIFIER ::= { saviMIB 2 }
saviCompliances OBJECT IDENTIFIER ::= { saviConformance 1 }

-- Compliance statements
saviCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
           "The compliance statement for entities which implement SAVI
            protocol.
           "
    MODULE
    MANDATORY-GROUPS {
        systemGroup,
        ifGroup,
        bindingGroup,
        filteringGroup
    }
    ::= { saviCompliances 1}

saviGroups OBJECT IDENTIFIER ::= { saviConformance 2 }

--Units of conformance

systemGroup OBJECT-GROUP
    OBJECTS {
        saviObjectsSystemMode,
```

```
        saviObjectsSystemMaxDadDelay,
        saviObjectsSystemMaxDadPrepareDelay
    }
    STATUS current
    DESCRIPTION
           "The system group contains objects corrsponding to savi system
            parameters.
           "
    ::= {saviGroups 1}

ifGroup OBJECT-GROUP
    OBJECTS {
        saviObjectsIfValidationStatus,
        saviObjectsIfTrustStatus,
        saviObjectsIfFilteringNum
    }
    STATUS current
    DESCRIPTION
           "The if group contains objects corresponding to the savi running
            parameters of each anchor.
           "
    ::= {saviGroups 2}

bindingGroup OBJECT-GROUP
    OBJECTS {
        saviObjectsBindingMacAddr,
        saviObjectsBindingState,
        saviObjectsBindingLifetime,
        saviObjectsBindingRowStatus
    }
    STATUS current
    DESCRIPTION
           "The binding group contains the binding
            information of anchor and soure ip address.
           "
    ::= {saviGroups 3}

filteringGroup OBJECT-GROUP
    OBJECTS {
          saviObjectsFilteringMacAddr
    }
    STATUS current
    DESCRIPTION
           "The filtering group contains the filtering
            information of anchor and soure ip address.
           "
    ::= {saviGroups 4}
END
```

9. Security Considerations

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. These are the tables and objects and their sensitivity/vulnerability:

- saviObjectsSystemTable - Unauthorized changes to the writable objects under saviObjectsSystemTable MAY disrupt allocation of resources in the network. For example, a device's SAVI system mode be changed by set operation to SAVI-DISABLE will give chance to IP source address spoofing.

- saviObjectsIfTable - Unauthorized changes to the writable objects under saviObjectsIfTable MAY disrupt allocation of resources in the network. For example, an anchor's ValidationStatus be changed by set operation to DISABLE will give chance to IP source address spoofing.

- saviObjectsBindingTable - Unauthorized changes to the writable objects under this table MAY disrupt allocation of resources in the network. For example, a static binding entry is inserted to the BST will give chance to IP source address spoofing.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- saviObjectsBindingTable, saviObjectsFilteringTable - The IP address and binding anchor information will be helpful to some attacks.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPsec), there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework (see [RFC3410], section 8), including full support for the SNMPv3 cryptographic mechanisms (for

authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

10. IANA Considerations

The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values recorded in the SMI Numbers registry:

Descriptor	OBJECT IDENTIFIER value
SAVI-MIB	{ ip XXX }

11. Contributors

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.

[I-D.ietf-savi-framework] Wu, J., Bi, J., Bagnulo, M., Baker, F., and C. Vogt, "Source Address Validation Improvement Protocol Framework", 2011.

[I-D.ietf-savi-dhcp] Bi, J.,Wu, J.,Yao, G., and F. Baker, "SAVI Solution for DHCP", 2011.

[I-D.ietf-savi-fcfs] E. Nordmark, M. Bagnulo, and E. Levy-Abegnoli, "FCFS SAVI: First-Come First-Serve Source-Address Validation for Locally Assigned IPv6 Addresses", 2011.

[I-D.ietf-savi-send] M. Bagnulo and A. Garcia-Martinez, "SEND-based Source-Address Validation Implementation", 2011.

12.2. Informative References

[RFC2223] Postel, J. and J. Reynolds, "Instructions to RFC Authors", RFC 2223, October 1997.

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.

[RFC4181] Heard, C., "Guidelines for Authors and Reviewers of MIB Documents", BCP 111, RFC 4181, September 2005.

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

[RFC4293] Routhier, S., "Management Information Base for the Internet Protocol (IP)", RFC 4293, April 2006.

12.3. URL References

[idguidelines] IETF Internet Drafts editor, "http://www.ietf.org/ietf/1id-guidelines.txt".

[idnits] IETF Internet Drafts editor,

"http://www.ietf.org/ID-Checklist.html".

[xml2rfc] XML2RFC tools and documentation, "http://xml.resource.org".

[ops] the IETF OPS Area, "http://www.ops.ietf.org".

[ietf] IETF Tools Team, "http://tools.ietf.org".

Appendix A. Change Log

From draft 00 to draft 01

- Change the value range of object saviObjectsSystemMode and add a new value savi-send(6).

Appendix B. Open Issues

Note to RFC Editor: please remove this appendix before publication as an RFC.

Authors' Addresses

Changqing An
CERNET
Network Research Center, Tsinghua University
Beijing 100084
China

Phone: +86 10 62603113
EMail: acq@cernet.edu.cn

Jiahai Yang
CERNET
Network Research Center, Tsinghua University
Beijing 100084
China

Phone: +86 10 62783492
EMail: yang@cernet.edu.cn

Jianping Wu
CERNET
Network Research Center, Tsinghua University
Beijing 100084
China

EMail: jianping@cernet.edu.cn

Jun Bi
CERNET
Network Research Center, Tsinghua University
Beijing 100084
China

EMail: junbi@cernet.edu.cn

Network Working Group
Internet Draft
Intended status: Standard Tracks
Expires: APR, 2012

J. Bi
J. Wu
Y. Wang
Tsinghua University
T. Lin
Hangzhou H3C Tech. Co., Ltd.
October 4, 2011

A SAVI solution for WLAN

draft-bi-savi-wlan-01.txt

Abstract

This document describes a source address validation solution for WLAN enabling 802.11i or other security mechanisms. This mechanism snoops NDP and DHCP to bind IP address with MAC address, and relies on the security of MAC address guaranteed by 802.11i or other mechanisms to filter IP spoofing packets. It can work in the special situations described in the charter of SAVI workgroup, such as multiple MAC addresses on one interface. This document describes three different deployment scenarios, with solutions for migration of mapping entries when hosts move from one access point to another.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 4, 2012.

Copyright Notice

Table of Contents

1. Introduction

This document describes a mechanism to perform per packet IP source address validation in WLAN. This mechanism performs ND snooping or DHCP snooping to bind allocated IP address with authenticated MAC address. Static addresses are bound to the MAC addresses of corresponding stations manually. Then the mechanism can check validity of source IP address in local packets according to the binding association. The security of MAC address is assured by 802.11i or other mechanisms, thus the binding association is secure.

The situation that one interfaces with multiple MAC addresses is a special case mentioned in the charter of SAVI. And this situation is the only special case that challenges MAC-IP binding. The mechanism to handle this situation is specified in the document.

There are three deployment scenarios specified in this document. The mechanism is deployed on different devices in different scenarios. The deployment detail is described in the document.

When hosts move from one access point to another, the migration of mapping entries may be triggered according to the specific mobility scenario. The mechanism to handle host mobility is specified in the document according to different deployment scenarios.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

3. IP-MAC Binding

This section specifies the operations of binding IP addresses to MAC addresses, and the clear of binding.

3.1. Data Structures

3.1.1. IP-MAC Mapping Table

This table maps IP addresses to corresponding MAC addresses. IP address is the index of the table. One IP address can only have one corresponding MAC address, while different IP addresses can be mapped to the same MAC address.

This table is used in control process. Before creating new IP-MAC bindings, this table must first be consulted in case of conflict in binding entries. This table must be synchronized with the MAC-IP table specified in Section 3.1.2.

The address allocated by DHCP has a limited lifetime, so the related entry has a limited lifetime, too. According to [RFC4862], stateless address also has a limited lifetime, the stations set this lifetime by itself.

3.1.2. MAC-IP Mapping Table

This table maps MAC addresses to corresponding IP addresses. MAC address is the index of the table. It is a one-to-many mapping table, which means a MAC address can be mapped to multiple IP addresses. Though multiple MAC addresses may exist on one interface, these MAC addresses must be mapped to different IP addresses.

This table is used for filtering and we will specify the details in Section 4. This table must be synchronized with the IP-MAC table specified in Section 3.1.1.

3.2. Pre-conditions for binding

In the binding based mechanism, the security of IP address is based on the security of the binding anchor. In WLAN, a number of security mechanisms on link layer make MAC address a strong enough binding anchor, for instance, 802.11i, WAPI, WEP.

If MAC address has no protection, attackers can spoof MAC address to succeed in validation. However, in general cases, if MAC address is not protected, more serious attack can be launched than IP spoofing attack.

3.3. Binding IP addresses to MAC addresses

All the static IP-MAC address pairs are configured into the IP-MAC Mapping Table with the mechanism enabled.

An individual procedure handles binding DHCP addresses to MAC addresses. This procedure snoops the DHCP address assignment procedure between attached hosts and DHCP server. DHCP snooping in WLAN is the same as wired network.

An individual procedure handles binding stateless addresses to MAC addresses. This procedure snoops Duplicate Address Detection procedure. ND snooping in WLAN is the same as wired network.

3.4. Clear Binding

Three kinds of events will trigger clearing binding:

1. The lifetime of an IP address in one entry has expired. This IP entry MUST be cleared.

2. A station leaves this access point. The entries for all the related MAC addresses MUST be deleted.

3. A DHCP RELEASE message is received from the owner of corresponding IP address. This IP entry MUST be deleted.

4. Source Address Validation

This section describes source address validation procedure on packet. In this procedure, all the frames are assumed to have passed the verifications of 802.11i or other security mechanisms.

This procedure has the following steps:

1. Extract the IP source and MAC source from the frame. Lookup the MAC address in the MAC-IP Mapping Table and check if the MAC-IP pair exists. If yes, forward the packet. Or else go to next step.

2. Lookup the IP address in the IP-MAC Mapping Table and check if the IP address exists. If no, insert a new entry into the IP-MAC Mapping Table and forward the packet. If yes, check whether The MAC address in the entry is the same as that in the frame. If yes, forward the packet. Else drop the packet.

5. Deployment Scenarios

This section specifies three deployment scenarios including two under centralized WLAN and one under autonomous WLAN. The deployment details and solutions for host mobility between access points are described respectively in each scenario.

5.1. Centralized WLAN

Centralized WLAN is comprised of FIT Access Points (AP) and Access Controllers (AC). In this scenario, this document proposes the following two deployment solutions.

5.1.1. AP Filtering

In this scenario, AC maintains IP-MAC Mapping Table while AP maintains MAC-IP Mapping Table. Packet filtering will be performed on each AP as specified in Section 4.

5.1.1.1. Candidate Binding

AP executes the procedure specified in Section 3.3. Candidate binding is generated after snooping procedure. Candidate binding must be confirmed by AC to be valid.

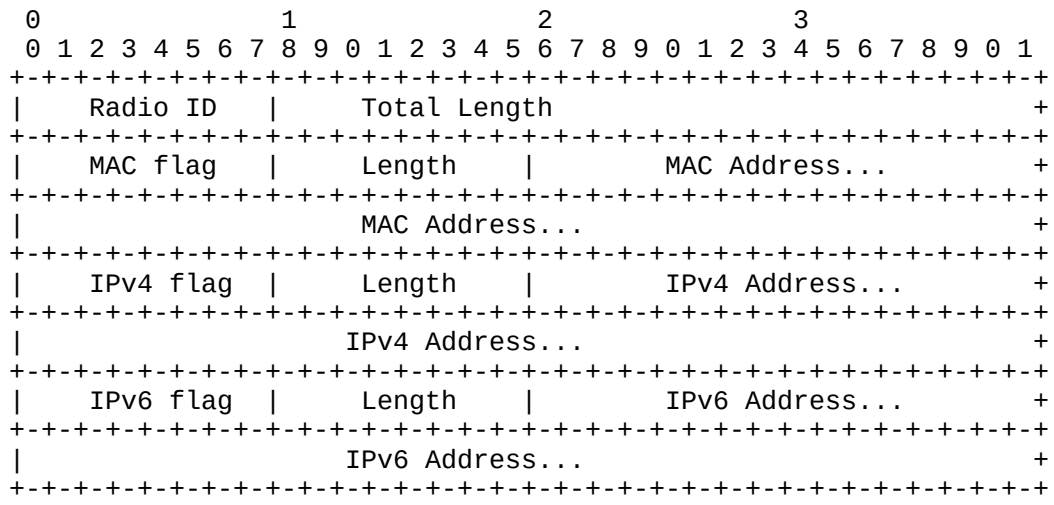
After a candidate binding is generated, AC is notified and checks whether the binding is valid or not. The validity of a candidate binding is determined if the binding does not violate any existing bindings in the IP-MAC Mapping Table. Otherwise if an address is not suitable for a host to use, AC notifies the corresponding AP. If the candidate binding is valid, AC adds an entry into the IP-MAC Mapping Table and notifies AP. Afterwards AP also adds an entry into the local MAC-IP Mapping Table.

5.1.1.2. CAPWAP Extension

CAPWAP protocol is used for communication between AP and AC. A new CAPWAP protocol message element is introduced, which extends the [CAPWAP]. The host IP message element is used by both AP and AC to exchange the binding information of hosts.

The host IP message element MAY be sent by AP. When AP generates a candidate binding, it reports the MAC address and related IP addresses to AC using this message, with suggestions of the state and lifetime of each IP address.

The host IP message element MAY be sent by AC. After AC checks the validation of a candidate binding, it replies using a message of the same format to inform AP the validation of each IP address with suggestions of its state and lifetime.



Radio ID: An 8-bit value representing the radio, whose value is between 1 and 31.

Total Length: Total length of the following fields.

MAC flag: An 8-bit value representing that the sub-field’s type is MAC address, whose value is 1.

Length: The length of the MAC Address field. The formats and lengths specified in [EUI-48] and [EUI-64] are supported.

MAC Address: A MAC address of the host.

IPv4 flag: An 8-bit value representing that the sub-field’s type is IPv4 address, whose value is 2.

Length: The length of the IPv4 Address field.

IPv4 Address: An IPv4 address of the host. There may exist many entries, and each entry is comprised of an IPv4 address, an 8-bit value for address state (only value 1 is used for now), and a 32-bit value for lifetime.

IPv6 flag: An 8-bit value representing that the sub-field’s type is IPv6 address, whose value is 3.

Length: The length of the IPv6 Address field.

IPv6 Address: An IPv6 address of the host. There may exist many entries, and each entry is comprised of an IPv6 address, an 8-bit value of address state (also one value for now), and a 32-bit value lifetime.

5.1.1.3. Mobility Solution

When a host moves from one AP to another, layer-2 association happens before IP packet transfer. Home AP deletes the binding when mobile host is disconnected, and foreign AP immediately requests the bound addresses with the associated MAC from AC. After AC tells AP the addresses should be bound, the binding migration is completed.

In WLAN, a host can move from an AC to another AC while keeping using the same IP address. To be compatible with such scenario, ACs must communicate to perform the binding migration.

CAPWAP extensions specified in Section 5.1.1.2 can also be used for communications between AC. The procedure of binding migration is the similar to that in the previous scenario. Home AC deletes the binding when mobile host is disconnected, and foreign AC requests the bound addresses with the associated MAC from Home AC.

5.1.2. AC Filtering

In this scenario, AC maintains both MAC-IP and IP-MAC Mapping Table and performs packet filtering. So all the packets must be firstly be forwarded to AC. AC executes the procedure specified in Section 3.3.

Mobility in one AC does not trigger any binding migration. Mobility between different ACs triggers binding migration and the procedure is the same as that in Section 5.1.1.3.

5.2. Autonomous WLAN

Autonomous WLAN is comprised of FAT Access Points. In this scenario, FAT AP maintains both MAC-IP and IP-MAC Mapping Table and performs packet filtering and executes the procedure specified in Section 3.3.

Mobility between different FAT APs will trigger binding migration, and the procedure is the same as that in Section 5.1.1.3.

6. Security Considerations

The security of address allocation methods matters the security of this mechanism. Thus it is necessary to improve the security of stateless auto-configuration and DHCP firstly.

7. IANA Considerations

There is no IANA Consideration currently.

8. Conclusions

This solution can satisfy the requirements of SAVI charter in WLAN enabling 802.11i or other security mechanisms.

9. Contributors

Guang Yao
Tsinghua University
Network Research Center, Tsinghua University
Beijing 100084
China
EMail: yaog@netarchlab.tsinghua.edu.cn

Yang Shi
Hangzhou H3C Tech. Co., Ltd.
Beijing 100085
China
EMail: rishyang@gmail.com

Hao Wang
Hangzhou H3C Tech. Co., Ltd.
Beijing 100085
China
EMail: hwang@h3c.com

10. Acknowledgments

This document was prepared using 2-Word-v2.0.template.dot.

11. References

11.1. Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[2] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

[3] IEEE 802.11i-2004: Amendment 6: Medium Access Control (MAC) Security Enhancements

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4862] Thomson, S., Narten, T. and Jinmei, T., "IPv6 Stateless Autoconfiguration", RFC4862, September, 2007.

[RFC3315] R. Droms, Ed., J. Bound, B. Volz, T. Lemon, C. Perkins, and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC3315, July, 2003.

[RFC5415] Control And Provisioning of Wireless Access Points (CAPWAP) Protocol Specification

11.2. Informative References

Authors' Addresses

Jun Bi
Tsinghua University
Network Research Center, Tsinghua University
Beijing 100084
China
EMail: junbi@cernet.edu.cn

Jianping Wu
Tsinghua University
Computer Science, Tsinghua University
Beijing 100084
China
EMail: jianping@cernet.edu.cn

You Wang
Tsinghua University
Network Research Center, Tsinghua University
Beijing 100084
China
EMail: wangyou@netarchlab.tsinghua.edu.cn

Tao Lin
Hangzhou H3C Tech. Co., Ltd.
Beijing 100085
China
EMail: lintaog@gmail.com

SAVI
Internet Draft
Intended status: Standard Tracks
Expires: May 2012

K.Xu, G.Hu, J.Bi, M.Xu
Tsinghua Univ.
F.Shi
China Telecom
November 20, 2011

The Requirements and Tentative Solutions for SAVI in IPv4/IPv6 Transition
draft-xu-savi-transition-00.txt

Abstract

SAVI Working Group is developing standardize mechanisms that prevent nodes attached to the same IP link from spoofing each other's IP addresses, and achieve IP source address validation at a finer granularity. Unfortunately, up to now, SAVI switch only works under the scenario of pure wire/wireless IPv6 Ethernet access subnet. In the current stage of IPv4/IPv6 transition which can't be cross over, SAVI has to make more progress to adapt it. This document describes the requirements and gives tentative solutions for the SAVI in IP4/IPv6 transition period. In RFC5565, Wu et.al proposal a softwire mesh framework to address the problem of routing information and data packets of one protocol how to pass through a single-protocol network of the other protocol. According to the real situation of CNGI-CERNET and China Telecom, document takes scenario of IPv4 packets transit IPv6 network into account.

Status of this Memo

Table of Contents

1. Introduction

Without a doubt, SAVI has made significant contribution for IP source address validation and anti-spoofing in the scenario of pure IPv6 Ethernet access subnet. Current situation is that IPv4 address has worn out but still takes a domination position for a long time, and IPv6 networking start to shrive. Meanwhile, SAVI switch only works at the scenarios of wire or wireless Ethernet, thus, there are lots of works have to do and many efforts need to make for adapting with the reality and promoting SAVI scheme. In the transition period from IPv4 to IPv6, approaches are classified into three types: dual stack, tunneling and translation. Regarding to real situation of CNGI-CERNET and China Telecomm which are the two of biggest Internet providers, this document mainly states the requirements and proposes some tentative solutions for scenarios of public 4over6[p4over6], lightweight 4over6[l4over6], which are the two implementations for scenario of IPv4-over-IPv6 in RFC5565. We hope that our proposal would be helpful for resolving problems of IP spoofing and validation in users' access subnet under transition period.

Public IPv4 over Access IPv6 Network (4over6) is a mechanism for bidirectional IPv4 communication between IPv4 Internet and end hosts or IPv4 networks sited in IPv6 access network. This mechanism follows the softwire hub and spoke model and uses IPv4-over-IPv6 tunnel as basic method to traverse IPv6 network. By allocating public IPv4 addresses to end hosts/networks in IPv6, it can achieve IPv4 end-to-end bidirectional communication between these hosts/networks and IPv4 Internet.

Public 4over6 can be generally considered as IPv4-over-IPv6 hub and spoke tunnel using public IPv4 address. Each 4over6 initiator will use public IPv4 address for IPv4-over-IPv6 communication. In the host initiator case, every host will get one IPv4 address; in the CPE (Customer premises equipment) case, every CPE will get one IPv4 address, which will be shared by hosts behind the CPE.

There is a slight different between public 4over6 and lightweight 4over6. Briefly, lightweight 4over6 mitigates IPv4 address exhaustion by sharing public IPv4 addresses amongst users, while public 4over6 host own a unique public IPv4 address. In lightweight 4over6 scenario, several hosts share a public address but have different port range by extending DHCPv4 and PCP protocol.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Requirements and solutions for SAVI in IPv4/IPv6 transition

In this section, we mainly talk about the requirements for SAVI in transition period regarding to public 4over6 and IVI approaches.

3.1. Public 4over6

Figure 1 illustrates the working scenario of public 4over6. Users in an IPv6 network take IPv6 as their native service. There are two types of users: dual-stack and CPE behind. Some users are end hosts which face the ISP network directly, while others are local networks behind CPEs, such as a home LAN, an enterprise network, etc. The ISP network is IPv6-only rather than dual-stack, which means that ISP can’t provide native IPv4 access to its users; however, it’s acceptable that one or more routers on the carrier side become dual-stack and get connected to IPv4 Internet. So if network users want to connect to IPv4, these dual-stack routers will be their entrances".

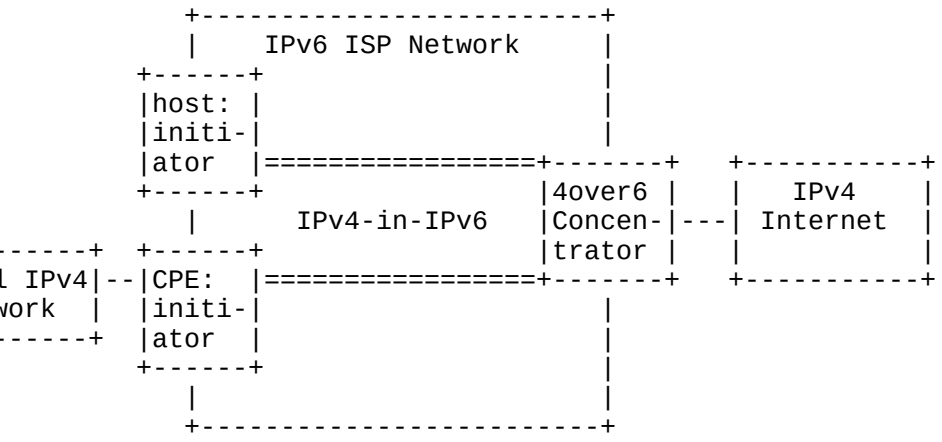


Figure 1 Public 4over6 scenario

Public 4over6 has stateful and stateless two working mode. Either stateful or stateless mode depends on whether it needs a mapping record for IPv4 and its corresponding IPv6 address in 4over6 Concentrator or not. The difference among them is that stateless mode

use the IPv6 address based on IPv4 embedded and initiator and concentrator both needs to parse/compose the address, while stateful mode means concentrator should restore the mapping records for IPv4/IPv6 address. Two types of users use DHCP or PCP protocol to retrieve IP address.

Two types of users multiple two types of working modes, that is four scenarios, we analyses them in detail.

a) Scenario 1:Dual-stack with stateful

For accessing IPv4 and IPv6 resources, this type of users owns IPv4 and IPv4 unrelated IPv6 addresses. Dual-stack hosts get their IPv6 address via DHCPv6 protocol as normal, however, IPv4 addresses allocation datagram for them need to encapsulate into tunnel, tunnel initiator is their IPv6 addresses and the 4over6 concentrator is the end of tunnel. For the reason of the toughness in access switch to parse IPv4 address from tunnel, SAVI switch only needs to snoop DHCPv6/PCP protocols and bind the relationship of <IPv6, MAC, Switch-Port>, however, 4over6 concentrator validates the mapping relationship of IPv4 to IPv6.

b) Scenario 2:Dual-stack with stateless

Even though this type of users has both IPv4 and IPv6 address, however, any of them could conduct to another. SAVI switch saves the relationship of <IPv6, MAC, Switch-Port > or <IPv4, IPv6, MAC, Switch-Port> would be ok.

c) Scenario 3: CPE-behind with stateful

In this scenario, hosts only have public IPv4 address and CPE plays the role of broker with dual-stack. SAVI switch should to snooping the DHCPv4/PCP protocols interaction and bind <IPv4, MAC, Switch-Port> relationship.

d) Scenario 4:CPE-behind with stateless

SAVI switch does the same thing with scenario C, the difference is that the start point of tunnel is initiated by CPE which use an IPv4-mapped IPv6 address.

In summary, SAVI switch should listen to the IP address allocation protocol like DHCPv6, DHCPv4, PCP etc. and bind host’s properties based on working scenario.

3.2. Lightweight 4over6

We no longer carry out a detailed description of each scenario in lightweight 4 over6 because there is nothing big changes compare with public 4over6 scheme. The difference exists in the way of host how to own an address. As mentioned before, public 4over6 host entirely own a unique public IPv4 address, while several lightweight 4over6 hosts share a public IPv4 address, but they have different port range. This change needs to extend DHCPv4[DHCPv6-map] and PCP protocol, thus, SAVI switch needs to listen to these address allocation protocols and bind relationship of <IPv4, MAC, Switch-Port, Port-range>.

4. Conclusions

There would be a long period from IPv4 to IPv6, public 4over6 is one of practical approach for inter-communication in the transition stage. SAVI switch focus on anti-snooping in users' access subnet by binding hosts' information. But till now, it only works at IPv6 environment. This document presents the SAVI requirements in this period, and in the meanwhile, we investigated working scenarios of public 4over6 in detail and gave some tentative solutions for SAVI adaption.

5. References

5.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5565] J.Wu, Y.Cui, C.Metz, E.Rosen, ''Softwire Mesh Framework'', RFC 5565, June 2009.

[p4over6] Y.Cui, J.Wu, P.Wu, C.Metz, O.Vautrin, Y.Lee, "Public IPv4 over Access IPv6 Network draft-cui-softwire-host-4over6-06", Internet-Draft, July 2011

[l4over6] Y.Cui, J.Wu, P.Wu, Q. Sun, C. Xie, C. Zhou, Y.Lee, " Lightweight 4over6 in access network draft-cui-softwire-b4-translated-ds-lite-04", Internet-Draft, Oct. 2011

[DHCPv6-map] T. Mrugalski, M. Boucadair, O. Troan, X. Deng, C. Bao, "DHCPv6 Options for Mapping of Address and Port draft-mdt-softwire-map-dhcp-option-01'', Internet-Draft, Oct. 2011

6. Acknowledgments

This document was prepared using 2-Word-v2.0.template.dot.

Authors' Addresses

Ke Xu
Tsinghua University
Department of Computer Science, Tsinghua University
Beijing, 100084
China
Email: xuke@mail.tsinghua.edu.cn

Guangwu Hu
Tsinghua University
Department of Computer Science, Tsinghua University
Beijing 100084
China
EMail: hgw09@mails.tsinghua.edu.cn

Fan Shi
China Telecom
Beijing Research Institute, China Telecom
Beijing 100035
China
EMail: shifan@ctbri.com.cn

Jun Bi
Tsinghua University
Network Research Center, Tsinghua University
Beijing 100084
China
Email: junbi@tsinghua.edu.cn

Mingwei Xu
Tsinghua University
Department of Computer Science, Tsinghua University
Beijing 100084
China
Email: xmw@csnet1.cs.tsinghua.edu.cn